

## Estimating System Level Costs

by Evin Stump  
Senior Consultant  
Galorath Incorporated

### Abstract

Cost estimating models often deal well with subsystem level costs, but not as well with integrating subsystems into systems. Today's complex systems usually involve integrating hardware and software, and systems must often operate in coordination with cooperating systems. Galorath Incorporated has recently improved the capability of its SEER-H (hardware) model and its SEER-SEM (software) model to better cope with these issues.

### Background

Over the last two decades, at Galorath Incorporated we kept abreast of the challenges of estimating software and hardware costs by continuous improvement of our SEER-SEM and SEER-H cost models. These models have been and today are effective tools for estimating software and hardware costs through the *program* level for software and through the *subsystem* level for hardware, under a wide variety of project conditions and situations.

The massive systems that are on drawing boards today and the even larger ones conceptualized for tomorrow present new challenges for realistic cost prediction. These systems are a highly interactive mixture of hardware and software, with software assuming ever-increasing responsibility for system success.

Increasingly, our clients have demanded more from our tools. Full system level costs are now demanded, with hardware and software fully integrated. Meeting this need has posed many challenges, ranging from defining nomenclature to obtaining credible data, to modeling issues. This paper reviews how we have met this challenge to date, and concludes by discussing where we believe this area of inquiry will go in the future.

### The Nomenclature Problem

One of the grand old men of cost analysis, Peter Korda, once said that you can't estimate the cost of anything if you don't know what it is. This seemingly trite and obvious statement is actually quite deep. Over the course of their careers, most cost analysts come to realize that it has profound implications.

One of the implications might be called the problem of cost taxonomy. Just as in the 1700s Carolus Linnaeus invented the system of biological classifications we still use today, and as hundred of others have since invented taxonomies for everything from airplanes to zymes, there needs to be a taxonomy for the proper classification of project costs.

In a sense, such a taxonomy exists, but it is not yet a controlled vocabulary that everyone interprets uniformly. A major point deeply in need of clarification is the distinction between subsystems, systems, and systems of systems, a commonly used hierarchy. They are all either at root the same thing with different names, depending on your point of view and where you happen to be standing relative to the hierarchy, or they are different things. If they are truly a hierarchy of different things, then different cost estimating treatments are likely to be needed at each level of the hierarchy. We believe they are distinctly different things.

While we at Galorath would welcome a major effort by all interested parties to clarify this subject once and for all, we were faced with the need to produce useful tools to do real world estimates. So we took the step of pragmatically defining them to the best of our understanding. Hopefully our approach will gain wide if not universal acceptance at least until better information is available.

Because at present we are focused on estimating system level costs, we began by considering the nature of a system. The notion of a system is inherently confusing because of our generally casual use of that word. One engineer's system may be thought of by another engineer as a subsystem. In particular, a subcontractor may have good reason to consider his product a system, while the prime contractor may have equally good reason to designate it a subsystem. Other terms of art that may contribute to the confusion are:

- Part
- Component
- Subassembly
- Assembly
- Unit
- Group
- Set
- Program

In our methodology we strive to avoid confusion by carefully defining and using only three terms that pertain to functional groupings. These apply to both software and hardware, and to combinations of them. They are:

- Work Element
- Subsystem

- System

We characterize these as follows:

**Work Element** – This is the fundamental estimating unit. It is something we have the means to directly estimate. (We seldom if ever have the means to directly estimate a system.) We recognize six types of work element and certain characteristic cost categories, namely:

- Rollup (a summation point for costs lower in a hierarchy—not truly a “work” element but an accounting convenience)
- Software (programs—development and maintenance costs)
- Electronic (printed circuit boards—development, production, and O&S costs)
- Mechanical (all structures and devices except printed circuit boards—development, production, and O&S costs)
- Site (a place where operations and support takes place—site characteristics help define O&S costs)
- Addin (a way to enter vendor quotes or other costs not needing to be estimated)

For the moment, we consider only the three types that directly produce parametric cost estimates for development and production, namely Software, Electronic and Mechanical.

Depending on circumstances, one might consider the estimate produced by such a work element to be for a part, a component, a subassembly, or even an assembly for hardware, or a program, module, component, CSCI, CSC, or CSU for software. To avoid potential confusion, in the system level cost scheme of things, we simply refer to it as a work element estimate. The estimate produced by SEER-H or SEER-SEM for a work element includes the cost of the element considered as an isolated standalone item; it also includes the cost of one or more work elements considered as a “subsystem,” as that term will now be defined.

**Subsystem** – In our approach, a subsystem can be designated one of two ways, as may appear most reasonable to the analyst. They are:

1. Any standalone work element plus all *interfacing entities* that may influence its costs at any time.
2. All of the children of any rollup element (unless designated as a system level rollup) plus all *interfacing entities* that may influence their costs at any time.

Most often, the interfacing entities are other work elements in the same project, but they can also be outside entities that are not included in the project's work element list.

In this scheme a work element may be a member of more than one subsystem. Also, every work element belongs to at least one subsystem.

The SEER-H parameter that most directly influences subsystem interface costs is called Subsystem Integration Level. This parameter enables description of the degree and complexity of an individual work element's integration with other work elements of the subsystem, and also with external entities if that is appropriate. An example of integration with an external entity is a sensor that provides feedback that is used to modify the state of an external object not listed in the SEER-H work breakdown structure. The more effort you have to put into making a given work element attach to, fit into, align with, coordinate with, work properly with, energize, calibrate, characterize, modify, or in any number of ways affect other work elements or external entities, the higher will be its subsystem integration costs.

In SEER-SEM, subsystem interface costs are controlled by a family of parameters under the general heading "Target Environment." Specific parameters are related to display requirements, memory constraints, time constraints, target system complexity, etc.

**System** – In any good dictionary, one can find five or more definitions of *system*. Some of the definitions are similar to the definitions given for *collection* or *group*. They do not emphasize the *purposefulness* of a system. In our approach, a project work file generally is created to estimate and assemble costs for one or more *purposeful* collections of work elements. A dictionary definition that accomplishes this fairly well is: "*Orderly combination or arrangement, as of parts or elements, into a whole; specifically, such combination according to some rational principle; any methodical arrangement of parts.*"

Even better from the standpoint of industry is this definition: "*Combination of several subsystems, sets, etc., which work together to perform one or more operational functions.*" Commonly in industry a system has several or all of the following aspects, not all of which are necessarily unique to systems:

- The components of a system may be physically separated. They are not necessarily all collocated as a single assembly. Subsystems, on the other hand, are generally visualized as comprising collocated elements.
- A system typically includes the support equipment that "touches" it. For example, an aircraft system is generally thought to include its ground support equipment.

- A system typically does not include the infrastructure that “supports” it. Roads, pipelines, utilities, etc., may be thought of as systems in themselves, but the industrial systems they support are generally not thought to include them.
- Generally a system can be characterized in terms of the mission (unique human purpose) it is designed to accomplish. For example, the mission of a launch vehicle is to transport objects into space.
- The mission associated with a system is not trivial and it is not merely a state of being.
- Multiple systems can operate cooperatively to form a system of systems that performs a sequence or set of collaborative missions. Example of cooperating systems:
  - Launch vehicle (system) – Transport a spacecraft into space
  - Spacecraft (system) – Operate in space to accomplish certain objectives
  - Ground station (system) – Collect information from one or more spacecraft for human use
- The same object can be regarded as a system in one context and as a subsystem in another. A subcontractor who builds an engine for a ship would likely consider it a system. He would estimate its cost by creating a group of work elements, estimating their standalone and work element interface costs, then summing the result. But he might recognize that these are not all of the costs he would incur to actually deliver the engine. There may be a need for complex alignments and calibrations, extensive testing, and permanent support equipment, along with added systems engineering and management effort. He would want to be able to include these costs. However, the prime contractor receiving the engine may well regard it as a subsystem. It would be simply a single work element along with certain interfaces in his scheme of estimating.

Given a system, what does system level cost (SLC) pay for? Our view is that it is the cost incurred to satisfactorily answer the following questions that have not been answered at lower levels of cost:

- Is the entire system configured according to plan and does it work as designed?
- Does the system work as designed with respect to cooperating systems and its mission?

SLC relates to effort, including both labor and material as appropriate, required to assure a satisfactory answer to these questions. Generally, this effort can be described under one or more of the following five SLC categories we have defined. These are very similar to established NASA and USAF categories of SLC with similar names, and the definitions we have adopted for them are also similar.

- System Engineering and Integration
- Integration, Assembly and Test
- System Program Management
- System Test Operations
- System Support Equipment

Of the above five categories, we assumed that only the first three are appropriate for production estimating. All five may be appropriate for development.

SLC may not in every case include all of the above cost categories. We have structured our approach so that users of our models can designate only the ones that apply.

In our approach SLC can be applied only at specifically designated rollup work elements. Most often, but not invariably, that is the total project element at the top of the work element structure.

In our current methodology, SEER-H is the focal point for system level costs. Since it does not estimate software costs, these must be linked from SEER-SEM. This is done using the addin element type.

### **Obtaining Credible Data**

We were fortunate that the NASA Independent Program Assessment Office (IPAO) made available to us the entire NAFCOM data base, and in addition provided some seed money to help us start our venture into system level cost. At first we aspired to collecting significant data directly from industry, but one by one, in spite of proffered non-disclosure agreements and NASA pleading, 18 commitments to send data melted down to only four as cautious financial managers reviewed the commitments. In addition to that, we already had a few useful data points that were at least partially germane to the subject.

The current (2002) NAFCOM database contains slightly over 120 projects comprising a variety of space objects, including orbiting satellites, interplanetary probes, manned spacecraft, tanks, engines, and the like. NASA suggested that we not use any projects earlier than 1975, so we removed a substantial part of the data for that reason. We also removed projects where parts of the data were missing. Our final analysis was based on 45 NAFCOM projects, the four projects submitted by industry, and a small number of projects we already had relevant data for.

Fortunately, NASA and the Air Force, in collecting and massaging the NAFCOM data, took some pains to thoughtfully segregate out the system level costs. We

did extensive data mining of these costs to try to find reasonable relationships within them and between them and other parameters of the projects. Many of the projects had written project histories, which were of some help in visualizing what may have caused costs to be what they were.

We found no relationships that could be confidently called cause and effect. We found just two strong and consistent correlations: System level costs in development were strongly correlated with subsystem level costs in development, and system level costs in production were strongly correlated with subsystem level costs in production.

Variations in the relationships between subsystem and system level costs we attributed to two factors, experience level of the project team, and complexity of the project, based largely on perceived inheritance from previous similar projects. We did this based on interpretation of project histories and expert opinion of experienced cost analysts.

The NAFCOM database does not break out software costs. They are included with the hardware costs for the hardware that contains the software. While it would have been helpful to have the software costs isolated, the very fact that they are there makes the database a reasonable basis for estimating system level costs for major projects that contain both software and hardware.

Although most of the data we used was for space hardware and software, we believe, based mainly on anecdotal evidence, that the results we obtained are also valid for other compact systems such as aircraft, ships, and ground stations. We are not as confident about the validity for highly distributed systems at this time, but continue to study that problem.

### **Modeling Issues**

We made an early decision that SLC should be applied only at rollup elements in the project cost breakdown structure. This seemed reasonable because subsystem level costs are already being rolled up and SLC costs were found to be strong functions of subsystem level costs, as previously noted.

However, clearly not every rollup element should be designated as system level. In a typical project, many cost rollups are done merely for accounting convenience. Therefore, it must be user choice to designate a rollup element as system level. How does the user decide?

We adopted the following criteria:

- SLC can be applied only at a rollup level work breakdown structure element. When applied, it takes into account *all* child work elements in every branch that leads to the rollup. It considers these work elements to be collectively designed to fulfill a common, non-trivial purpose, preparation for which requires incurring costs above and beyond their standalone costs and the costs of creating the appropriate interfaces for the individual elements.
- If the optional SLC estimating capability is not used, the only costs produced are the standalone costs of the individual work elements, plus for each work element the cost of creating the appropriate interfaces between it and all other work elements and outside entities. Note that parameters controlling subsystem integration costs typically will add less than twenty percent to the cost of an individual work element. The optional SLC feature, on the other hand, can add much more than that to the total cost of a group of work elements that are designated as a system, depending on circumstances. The added cost depends on the cumulative subsystem cost, the complexity of the system, and the experience of the project team.
- As previously noted, the same object can be regarded as a system in one context and as a subsystem in another.
- More than one system can be designated in a single project file. Any rollup element can be designated as system level, without restriction. However, care must be used in creating a system below another system, and especially in creating more than one system below another system. This in effect creates a “system of systems.” At the present state of the art of our modeling, systems of systems can be difficult to characterize properly. Also, the database underlying the SLC feature is based on discrete systems, not systems of systems. Use of the SLC feature to estimate systems of systems is currently problematic, unless there is confidence in doing it based on calibrations derived from past projects.
- The SLC feature can estimate any or all of the following system level cost categories in the development phase of a project:
  - System Engineering and Integration
  - Integration, Assembly & Test
  - System Program Management
  - System Test Operations
  - System Support Equipment

The first three in the above list also can be separately estimated for the production phase of a project. If it is known that a system level estimate is appropriate, but that one or more of the above cost categories is not, one or more of them can be selectively turned off. Reasons why a particular category might be turned off include:

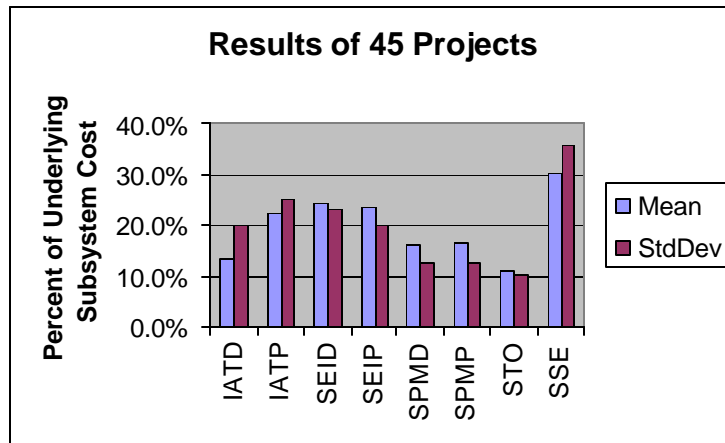
- The system level effort is known to be negligible

- The costs are deemed already accounted for at subsystem level
- The cost is already covered or included elsewhere, for example when system support equipment from a previous project can be reused or has been entered as an Addin cost element based on a vendor quote.

## Findings

Our most useful results are summarized in the chart below. In this chart, please interpret the labels on the horizontal axis as follows, *all at system level*:

- IATD – Integration, assembly and test, development
- IATP – Integration, assembly and test, production
- SEID – System engineering and integration, development
- SEIP – System engineering and integration, production
- SPMD – System program management, development
- SPMP – System program management, production
- STO – System test operations
- SSE – System support equipment



Our current SLC model accounts for the variability (standard deviation) by means of parameters by which the user can define the expected average experience of the project team in both development and production, and also the “complexity” of the project. These parameters are separately defined for each of the eight cost categories in the chart above.

Complexity is a commonly used parameter in cost models, probably because it conveys a strong intuitive message of complicatedness and possible difficulty, and hence increased cost. However, it is one thing to cite complexity as a parameter, and another thing entirely to convey to the user a clear idea of how to select a value for it. To solve this, we put complexity on a fourteen-point scale ranging from VERY LOW- to VERY HIGH. At five levels in the scale, we placed descriptive information to help the user arrive at a reasonable parameter setting. The descriptive information necessarily varies from one SLC cost category to another. The descriptions are too lengthy to provide here, but as an example here is the information we developed for setting the system program management complexity:

*This parameter measures the perceived structural inherent risk in the project, and the likelihood and severity of the impacts if risk drivers on the project plan.*

VERY HIGH	Project goals are unclear or unrealistic, and likely to be volatile. Activities of organizations that are widely scattered and/or culturally different must be coordinated. Interference from or non-cooperation of some stakeholders likely. Planning very difficult because of poor access to information.
HIGH	Project goals are challenging but deemed doable. Goals are likely to be mostly stable but some may change. Activities of organizations that are widely scattered and/or culturally different must be coordinated. Stakeholders will be mostly cooperative. Planning somewhat difficult because of poor access to information.
NOMINAL	Project goals are deemed reasonable and doable. Little change in goals is expected. Most of the project team is collocated. Most information needed for planning is readily available.
LOW	Project goals are not at all challenging. No significant change in goals is expected. The entire team is collocated. Stakeholders will be cooperative because of high interest in project success. Virtually all information needed for planning is readily available.
VERY LOW	Project goals are easily accomplished and stable. The entire project team is collocated. Stakeholders will make every effort to clear the way for rapid and complete project success. All information needed for planning is readily available.

### **For the future**

We continue to seek additional data, hard or anecdotal, about the distinctions that should be made between subsystems and systems and the relationships between system level costs and other aspects of the system. As we learn more, we will update our modeling approach accordingly.

At this time we are particularly interested in possible differences between compact and distributed systems with respect to system level cost. We welcome any and all suggestions for clarifying these possible differences.

Some potential users of our SLC model have inquired about its potential use in estimating systems of systems. We believe this type of estimating capability is a real need even today, and will become more so in the future. Unfortunately, little credible data has been available up until now, and we recommend against using the model in this way unless the user can realistically calibrate it using well established calibration procedures. As experience with large systems and systems of systems grows, these problems will likely be resolved.